A Comparison of Techniques for Name Matching

Taoxin Peng, Lin Li, Jessie Kennedy
School of Computing
Edinburgh Napier University
Edinburgh, UK
{t.peng, l.li, j.kennedy}@napier.ac.uk

***Abstract*—Information explosion is a problem for everyone nowadays. It is a great challenge to all kinds of businesses to maintain high quality of data in their information applications, such as data integration, text and web mining, information retrieval, search engine, etc. In such applications, matching names is one of the popular tasks. There are a number of name matching techniques available. Unfortunately, there is no existing name matching technique that performs the best in all situations. Therefore, a problem that every researcher or a practitioner has to face is how to select an appropriate technique for a given dataset. This paper analyses and evaluates a set of popular name matching techniques on several carefully designed different datasets. The experimental comparison confirms the statement that there is no clear best technique. Some suggestions have been presented, which can be used as guidance for researchers and practitioners to select an appropriate name matching technique in a given dataset.**

***Index Terms*—name matching, duplicate, data integration, data cleaning**

I. INTRODUCTION

Information explosion is a problem for everyone nowadays. It is a great challenge to all kinds of businesses to maintain the high quality of data. There are many reasons for a business to fail. One of them is the poor data quality in the information system an organization has. If the data is not clean, the queries and reports generated in the system will be wrong, in which case directors/managers will either make wrong decisions or mistrust the reports and not make any decisions on them. This simply puts the value of the information system in question. Therefore, to be able to benefit from an information system, the data stored must have high quality. The higher the data quality is in the system, the better chance the business will have to secure a success.

When data need to be integrated from multiple sources, such as comprehensive information systems, data warehousing applications, it always has a problem: how to identify data records that refer to equivalent entities, which is called a duplicate problem. For example, considering a person's name in the name record, a same person can be represented as "John Smith" in one data source, while as "John Smtih" in another data source. It might introduce a duplicate error if these two entities are not treated as the same one when integrate the data from these two sources. Such kind of duplicate problems is also called the name matching problem. Generally speaking, name matching deals with the problem of whether two name strings refer to the same name. There are several reasons for databases to have such a problem, which include typos during data entry, variations in representation of names, etc., especially when multiple data sources need to be integrated, e.g., in data warehouses.

Matching names in databases has been a persistent and well-known problem for years [1]. There are several techniques available that deal with the problem [2, 3, 4, 5, 6, 7]. However, since there is no clear best technique for all kinds of datasets [8], a problem still existing for researchers and practitioners is how to select a technique for a given dataset [9]. In past decade, several researchers have challenged this problem [8, 10, 11, 12, 13]. However, none of them have done such a comprehensive analysis and comparison work that is done in this paper. The contributions of this paper are to overview five popular character-based name matching techniques, evaluate whether the following factors will have effect on the performance: the error rate in a dataset, the threshold value, the selected type of strings in a dataset, the type of typos, i.e. typos occur at different part in a string and the size of a dataset, by using 42 carefully designed datasets.

The rest of this paper is structured as follows. Related works are described in next section. Section III introduces techniques that will be examined. The main contribution of this paper is presented in section IV that describes the preparation of the datasets, the experiments, the analysis and comparisons. Finally, this paper is concluded and future work pointed out in section V.

II. RELATED WORKS

Bilenko *et al* [10] and Cohen *et al* [11] evaluated and compared a set of existing string matching techniques, which include popular character-based techniques, token-based techniques and hybrid techniques. They claimed that the Monge-Elkan performed best on average and SoftTF-IDF performed best overall. However, their works did not consider the effect of the error rate, the type of typos and the size of a

dataset on the performance. Besides, regarding the threshold value used for matching, their work only mentioned that a suitable threshold value was chosen, but not mentioned how and whether or not this value was universal for all considered techniques.

Peter Christen [8] thoroughly discussed the characteristics of personal names and the potential sources of variations and errors in them, and also evaluated a number of commonly used name matching techniques, considering given names, surnames and full names separately, and proposed nine useful recommendations for technique selection when dealing with name matching problems. Particularly, the author pointed out the importance of choosing a suitable threshold value. It was argued that it was a difficult task to select a proper threshold value and even small changes of the threshold could result in dramatic drops in matching quality. However, similar to Cohen *et al*'s work, the author did not consider any effect of the error rate, the type of typos and the size of a dataset.

Hassanzadeh *et al* [12] presented an overview of several string matching techniques and thoroughly evaluated their accuracy on several datasets with different characteristics and common quality problems. The work was focused only on token-based string matching techniques. The effect of types of errors and the amount of errors were both considered. Types of errors considered include edit errors, token swap and abbreviation replacement. It was claimed that the threshold value used for the matching task would influence the individual performance of matching techniques. However the type of typos and the size of datasets were not considered.

Recently, Peng *et al* [13] presented an evaluation work on techniques for name matching. The work considered a variety of factors, such as the error rate, the size of a dataset, which might have effect on the performance of such techniques. Their preliminary experimental results confirmed that there is no overall clear best technique, suggesting that in general Jaro-Winkler and Jaro perform better that others in matching names. The work also claimed that the error rate in the dataset has effect on threshold values. However, they didn't consider types of typos and first names.

III. MATCHING TECHNIQUES

Name matching can be defined as "the process of determining whether two name strings are instances of the same name" [14]. To deal with name matching, there are mainly two types of matching techniques: character-based and token-based techniques. Character-based similarity techniques are designed to handle well typographical errors. However, it is often the case that typographical conventions lead to rearrangement of words e.g., "John Smith" vs. "Smith, John". In such cases, character-based techniques fail to capture the similarity of the entities. Token-based techniques are designed to compensate for this problem [7]. Therefore, character-based similarity techniques are good for the single word problem, while token-based for the matching with more than one word. Since our experiments are focused on single names, we evaluated five popular character-based techniques: Levenstein, Smith-Waterman, Jaro, Jaro-Winkler and Q-Gram.

A. Levenshtein

The Levenshtein distance [4] is defined to be the minimum number of edit operations required to transform string s_1 into s_2. Edit operations are *delete*, *insert*, *substitute* and *copy*.

The Levenshtein similarity measure can be calculated by:

$$Leveshtein(s_1,s_2) = 1.0 - \frac{dist(s_1,s_2)}{max\ (|s_1|,|s_1|)}$$

where dist(s_1, s_2) refers to the actual Levenshtein distance function which returns a value of 0 if the strings are the same or a positive number of edits if they are different. The value of such a measure is between 0.0 and 1.0 where the bigger the value, the more similar between the two strings.

B. Smith-Waterman

This algorithm is based on a dynamic programming approach similar to Levenshtein distance, but allows gaps as well as character specific match scores [2]. Let *t* being the final best score obtained based on the dynamic programming matrix and g being the match score value. In this paper, smith-waterman similarity measure between two strings s_1 and s_2 is calculated by:

$$Smith - Waterman(s_1,s_2) = \frac{\mathrm{t}}{\min(|s_1|,|s_1|) \times \mathrm{g}}$$

C. Jaro

Jaro [3] introduced a string comparator that accounts for insertions, deletions and transpositions, which was mainly used for comparison of first and last names [7].

Given strings $s = s_1 \ldots s_k$ and $t = t_1 \ldots t_l$, define a character s_i in *s* to be common with *t* iff there is a $t_j = s_i$ in *t* such that $i\text{-}H \leq j \leq i+H$, where $H = \min(|s|, |t|)/2$. Let $s' = s'_1 \ldots s'_k$ be the characters in *s* which are common with *t* (in the same order they appear in *s*) and let $t' = t'_1 \cdots t'_l$ be the same in *t*. A transposition for s', t' is a position i such that $s'_i \neq t'_i$. Let $T_{s',t'}$ be half the number of transpositions for s' and t'. Jaro similarity measure for string *s* and *t* is calculated by:

$$Jaro(s,t) = \frac{1}{3}\left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - T_{s',t'}}{|s'|}\right)$$

D. Jaro-Winkler

William Winkler proposed a variant of the Jaro metric based on empirical studies that fewer errors typically occur at the beginning of names [6]. Jaro-Winkler similarity measure between string s_1 and s_2 is calculated by:

$$Jaro - Winkler(s_1,s_2) = Jaro(s_1,s_2) + \frac{p}{10}(1 - Jaro(s_1,s_2))$$

where p=max(q, 4), and q is the longest common prefix of two strings s_1 and s_2.

E. Q-Gram

The Q-gram metric is based on the intuition that two strings are similar if they share a large number of common q-grams. Q-grams are sub-strings of length q [5]. Let $G_q(s)$ denote all the q-grams of a string s obtained by sliding a window of length q over the characters of s. The q-gram similarity measure between strings s_1 and s_2 is calculated by:

$$q-gram(s_1,s_2)=\frac{|G_q(s_1)\cap G_q(s_2)|}{max\ (|G_q(s_1)|,|G_q(s_2)|)}$$

IV. EXPERIMENTS AND EVALUATION

In our experiments, we focus on the performance of the above five popular string matching techniques on two types of strings, i.e., last name strings and first name strings. For last name strings, 8 different sizes of datasets ranging from 200 records to 9454 records are used, while a 2300 record dataset for first name strings. In this paper, the error rate of a dataset is defined as the ratio of erroneous records and the whole number of records in the dataset. There are three error rates considered: low, medium and high with values of about 20%, 50% and 70% respectively. For each size, three datasets with different error rates are used.

Regarding the type of strings, only first name and last name strings are considered. We also consider three different types of typos in strings: typos occurring at front, middle and end of a string respectively. The datasets designed for such experiments only have 2300 records and have typos occurring either at all three parts, or only at the front part, or only at the end part of a string.

A. Datasets Preparation

In the absence of common datasets for data cleaning, we prepare our data for experiments as follows.

With respect to last names, the datasets are based on a historical set of real Electoral Roll data. First, a one million record dataset was extracted, from which a personal last name list was created. This list contains 9454 clean, non-duplicate personal last names. Then, a last name dataset is generated, which contains these 9454 last names, with an ID number associated to each of the records.

Erroneous records were created by doing the following four operations manually to the name field of records: inserting, deleting, substituting and replacing characters. There were in total twenty-four datasets generated and the number of records for these last name datasets ranges from 200 to 9454. For each size, there are three datasets generated having a different error rate associated with. For example, the following table summarized the last name datasets associated with low error rate used for the experiments.

TABLE I
LAST NAME DATASET WITH LOW ERROR RATE

Datasets	Error Rate
9454 Records	Low
7154 Records	Low
5000 Records	Low
3600 Records	Low
2300 Records	Low
1000 Records	Low
500 Records	Low
200 Records	Low

Similar to the generation of last name datasets, a dataset containing 2300 clean, non-duplicate first names is created. The evaluation of the effect of the type of typos on performance is done by the experiment on three groups of first name datasets and three groups of last name datasets, which all include 2300 records with an error rate associated. There are total 18 datasets generated for this experiment. For example, table 2 and 3 summarized such datasets associated with high error rate:

TABLE II
FIRST NAME DATASETS WITH DIFFERENT TYPE OF TYPOS

First name Dataset	Error Rate	Type of typo
2300 Records	High	Three parts
2300 Records	High	Front part
2300 Records	High	End part

TABLE III
LAST NAME DATASETS WITH DIFFERENT TYPE OF TYPOS

Last name Dataset	Error Rate	Type of typo
2300 Records	High	Three parts
2300 Records	High	Front part
2300 Records	High	End part

B. Measures

A target string is a *positive* if it is returned by a technique; otherwise it is a *negative*. A positive is a *true positive* if the match does in fact denote the same entity; otherwise it is a *false positive*. A negative is a *false negative* if the un-match does in fact denote the same entity; otherwise it is a *true negative*.

We evaluate the matching quality using the F-measure (F) that is based on precision and recall:

$$F=\frac{2\times P\times R}{P+R}$$

with P (precision) and R (recall) defined as:

$$P=\frac{|true\ positives|}{(|true\ positives|+|false\ positives|)}$$

$$R=\frac{|true\ positives|}{(|true\ positives|+|false\ negatives|)}$$

Clearly, a trade-off between recall and precision exists, if all targets are matched, recall will be 100% but precision will be low. Conversely if precision is high, recall will be low. F-

measure is a way of combining the recall and precision into a single measure of overall performance [15]. In our experiments, precision, recall and F-measure are measured against different value of similarity thresholds, θ. For the comparison of different techniques, the maximum F-measure score across different thresholds is used.

C. Results and Evaluation

In this section, testing results for both last name and first name datasets are analysed and evaluated based on the accuracy and timing performance of the five selected techniques.

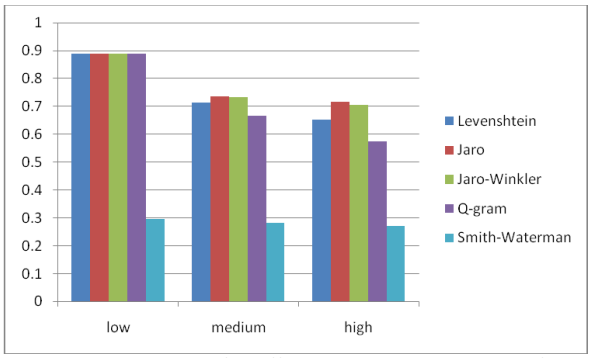


Fig. 2: Maximum F score for different techniques on datasets of 3600 records with three different error rates

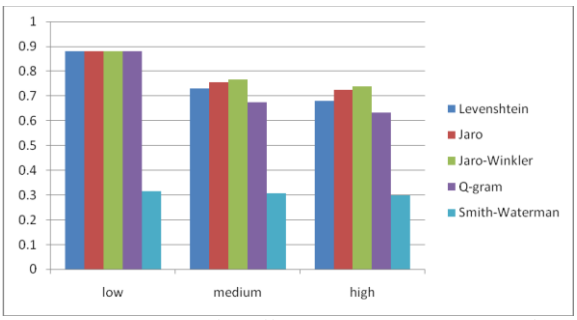


Fig. 3: Maximum F score for different techniques on datasets of 7154 records with three different error rates

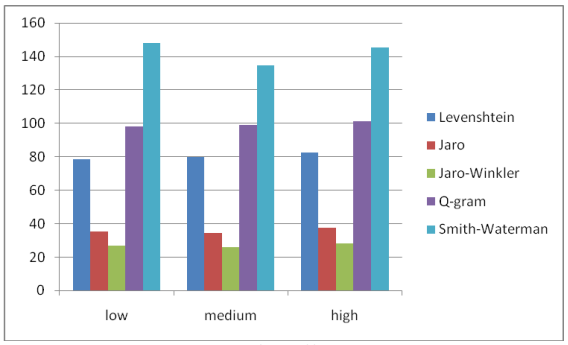


Fig. 4: Time used (in seconds) for different techniques on datasets of 7154 records with three different error rates

Without the consideration of types of typos

Testing results without the consideration of types of typos in strings are analysed. The experiments are on the 24 datasets with records ranging from 200 to 9454 separately. Results show that in general, the size of a dataset is not sensitive to the accuracy relative to the threshold values when the size of the dataset is equal to or more than 1000, except Smith-Waterman. When the size is smaller than 1000, the best F-score is sensitive to the value of thresholds. Fig. 1 represents the results from datasets of 7154 records. It shows the accuracy relative to the value of thresholds on different datasets with different error rates. For all graphs, the horizontal axis is the value of threshold.

1) Effect of Error Rates on Threshold Values: As shown in graphs in Fig. 1, for all techniques, the higher the error rate in the dataset, the lower the threshold value is required in order to achieve the best performance. For example, Levenshtein achieves the best F score over datasets with the high error rate at threshold 0.8, while it achieves the best over the datasets with medium and low error rate at threshold 0.85 and 0.99 respectively when the size of a dataset is equal to or more than 1000. Jaro-Winkler is less sensitive and works well on datasets with both medium and high error rate at threshold 0.95 when the size of a dataset is 500 or more.

2) Effect of Error Rates on Performance: Experimental results towards all eight groups of last name datasets show that in general, all five techniques perform better on datasets with a lower error rate. For example, Fig. 2 and 3 show the performance (F-measure) of all five techniques on datasets of 3600 and 7154 records with three different error rates, respectively. Leveshtein, Jaro, Jaro-Winkler and Q-gram perform equally the best among the five techniques on datasets with the low error rate. When the error rate is increased, the performance of techniques varies on different error rates. For example, Fig. 2 shows that the performance of all five techniques is in decreasing along with the increasing of the error rate. Looking at performance of individual techniques, Jaro performs the best, slightly better than Jaro-Winkler and Levenshtein on datasets of 3600 records with the medium and high error rate. However, Fig. 3 shows that Jaro-Winkler performs the best on datasets of records 7154 with the high error rate. Overall, Smith-Waterman performs the worst of all the five techniques. The effect of size will be further discussed in next section.

3) Effect of the Size of Datasets on Performance: As mentioned at the beginning of this section, the change of the size of a dataset is not sensitive to the acccuracy relative to the threshold except for Smith-Waterman when the size of the dataset is equal to or more than 1000. For example, Smith-Waterman achieves the best F score over datasets of 7154 records with high error rate at threshold 0.9, while it achieves the best over datasets of 3600 records with high error rate at threshold 0.85. However, the effect of the size of datasets on performance is significant when the size of a dataset is smaller. Table 4 summarises the comparison of performance among the five techniques in different datasets in a decending order.

This table shows that with the medium or high error rate, the performance of Jaro and Jaro-Winkler is better when the size of a dataset is more than 2300, while Levenstein is better

TABLE IV
A COMPARISON OF PERFORMANCE WITH SIZE CONSIDERED

Low	Medium	High	Data Size
LE=J=Q>JW>SW	J>JW>LE>Q>SW	JW>LE>J>Q>SW	9454
LE=J=Q>JW>SW	JW>J>LE>Q>SW	JW>J>LE>Q>SW	7154
LE=J=Q>JW>SW	JW>J>LE>Q>SW	JW>J>LE>Q>SW	5000
LE=J=Q>JW>SW	J>JW>LE>Q>SW	J>JW>LE>Q>SW	3600
LE=J=Q>JW>SW	J>LE>JW>Q>SW	JW>J>LE>Q>SW	2300
JW>LE=J=Q>SW	J>LE>JW>Q>SW	JW>LE>J>Q>SW	1000
JW=LE>J=Q>SW	JW>LE>J>Q>SW	J>JW>LE>Q>SW	500
JW=LE=J=Q>SW	LE>Q>J>JW>SW	LE>Q>J>JW>SW	200

with lower error rate when the size of a dataset is equal to and more than 2300. Levenstein is also the best on datasets with the medium and high error rate when the size is 200. Jaro-Winkler becomes the best on datasets with a low error rate when the size is from 200 to 1000.

4) Effect on Timing, As shown in Fig 4, in general, Jaro-Winkler costs the least time among the five algorithms while Smith-Waterman costs the most time. The time used by Jaro is slightly more than that by Jaro-Winkler, and much better than the other three. Our experiments results agree that smaller datasets cost less time. However, the effect of error rates on timing is not significant.

With the consideration of the type of typos:

Testing results with the consideration of the type of typos in strings are based on experiments o n two sets of datasets. One set contains three groups of first name datasets with a size of 2300. The other set contains three groups of last name datasets with the same size of 2300. Each group contains datasets associated with a predefined error rate. See table 2 and 3.

Results show that in general, the effect of error rates on threshold value selection and performance are the same as the testing results of the previous last name datasets. That is, the higher the error rate in a dataset, the lower the threshold value is required in order to achieve the best performance, and techniques perform better in lower error rate datasets. Further comparisons with last name datasets indicate that the performance varies based on different types of typos, i.e., typos occurring at different positions of a string.

1) Effect of the Type of Typos on Performance: Experimental results show that Jaro and Jaro-Winkler techniques are sensitive to the type of typos within a name stirng. Fig.5 and Fig.6 show the performance of the five techniques on first name datasets when typos occur at the front and the end part of a string respectively. It is clear to see that Jaro-Winkler and Jaro perform better when typos occur at the end part of a string, while Levenshtein performs better when typos occur at the front part of a string.

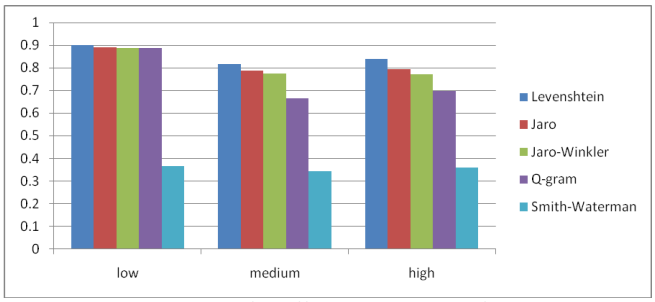


Fig. 5: Maximum F score for different techniques on first name datasets of 2300 records when typos occur at the front part of a string.

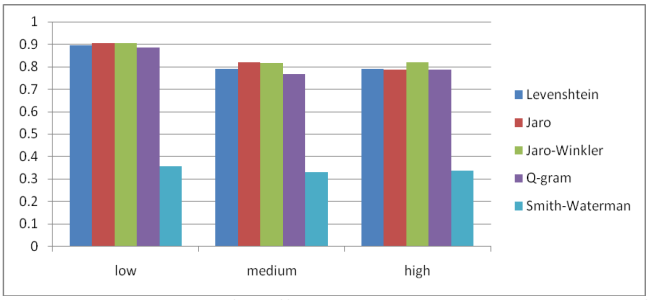


Fig. 6: Maximum F score for different techniques on First name datasets of 2300 records when typos occur at the end part of a string.

2) Comparison of the Effect of the Type of Strings on Performance: Fig. 7 and Fig. 8 show performances of all five techniques on both first and last name datasets of 2300 records with three different error rates when typos occur at the front and the end part of a string respectively. Results show that in general, techniques perform better on first name datasets than last name datasets except Smith-Waterman that performs better on last name datasets when typos occur at the end part of a string. For first name datasets, Levenshtein performs significantly better when typos occur at the front part of a string. The figures also show that the difference of performance among the five techniques is more significant when the error rate is medium or high.

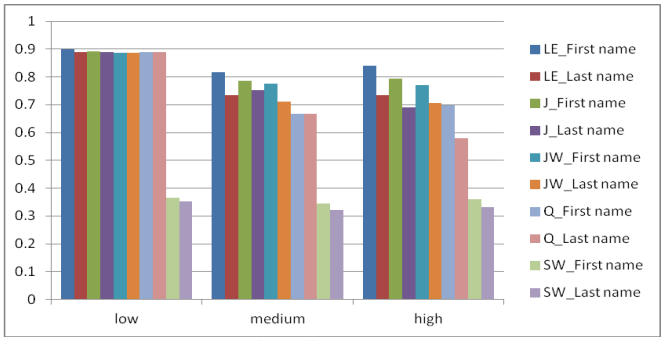


Fig. 7: Maximum F score for different techniques on First name and Last name datasets of 2300 records when typos occur at the front part of a string.

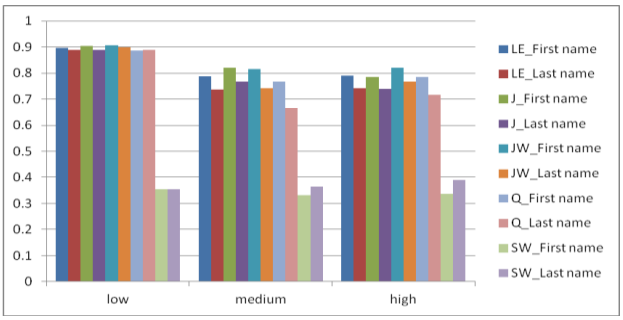


Fig. 8: Maximum F score for different techniques on First name and Last name datasets of 2300 records when typos occur at the end part of a string.

V. CONCLUSION AND FUTURE WORK

This paper has analysed and evaluated five popular character-based name matching techniques. A comprehensive comparison of the five techniques has been done based on a series of experiments on different last name and first name datasets. The comparison results confirmed the statement that there is no clear best technique. The size of datasets, the error rate in datasets, the type of strings in a dataset and the type of typos in a string all have significant effect on performance of these five techniques. In general, Jaro-Winkler and Jaro perform better than others, especially on datasets with a higher error rate associated. This agrees with the statement that they mainly used for comparison of first and last names [7]. However, with a low error rate associated, Leveshtein, Jaro, Jaro-Winkler and Q-gram perform equally the best. Considering the type of strings, i.e., last name strings or first name strings, the experiments on 2300 record datasets show that techniques perform better on first name strings, in general. Whether this pattern is independent from the size of a dataset or not, it requires further investigation. The error rate also has effect on threshold values. The higher the error rate in the dataset, the lower the threshold value is required in order to achieve the best performance. Time used by these techniques on different datasets has also been analysed and compared. Overall, Jaro-Winkler and Jaro are significantly faster than others. Therefore, it is suggested that the selection of a technique should depend on the nature of a dataset.

The work introduces a number of further investigations, including: 1) to do a comparison of the effect of the type of strings on more different sizes of datasets; 2) to do similar experiments on popular token-based string matching techniques, especially to evaluate whether the size of a dataset has effect on performance or not; 3) to do further analysis in order to evaluate whether there is a method to select a threshold value for any of the matching techniques on a given dataset.

RERENRENCES

[1] J. Hermansen, "Automatic Name Searching in Large Databases of International Names", Georgetown University Dissertation, Washington, DC, 1985.

[2] T. Smith and M. Waterman, "Identification of Common Molecular Subsequences", J. Mol. Biol., Vol. 147, pp.195-197, 1981.

[3] M. Jaro, "Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa,Florida", Journal of the American Statistical Associations, Vol. 89, pp. 414-420, 1989.

[4] G Navarro, "A guided tour to approximate string matching", ACM Computing Surveys, Vol. 33, No. 1, pp.31–88, 2001.

[5] E. Ukkonen, "Approximate string matching with q-grams and maximal matches", Theoretical Computer Science, Vol. 92, No. 1, pp. 191-211, 1992.

[6] W. Winkler, "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage", Proceedings of the Section on Survey Research Methods, American Statistical Association, 354-359, 1990.

[7] A. Elmagarmid, P. Ipeirotis, and V. VeryKios, "Duplicate Record Detection: A Survey", IEEE Trans. Knowl.Data Eng., Vol.19, No.1, pp. 1-16, 2007.

[8] P. Christen, "A Comparison of Personal Name Matching: Techniques and Practical Issues", Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops (ICDMW '06). IEEE Computer Society, Washington, DC, USA, pp.290-294, 2006.

[9] L. Li, T. Peng, and J. Kennedy, "A Rule Based Taxonomy of Dirty Data", GSTF International Journal on Computing, Vol. 1, No. 2, 2011.

[10] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, "Adaptive Name Matching in Information Integration", IEEE Intelligent Systems, vol. 18, no. 5, pp. 16-23, 2003.

[11] W. Cohen, P. Ravikumar, and S. Fienberg, "A comparison of string distance metrics for name-matching tasks", Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web, pp.73-78, 2003.

[12] O. Hassanzadeh, M. Sadoghi, and R. Miller, "Accuracy of Approximate String Joins Using Grams", Proceedings of QDB'2007, pp.11-18, 2007.

[13] T. Peng, L. Li and J. Kennedy, "An Evaluation of Techniques for Name Matching", Proceedings of Annual International Conference on Data Analysis, Data Quality and Metadata Management, Singapore, June, 2011

[14] F. Patman and P. Thompson, "Names: A New Frontier in text Mininin" ISI-2003, Springer LNC 2665, p. 27-38, 2003.

[15] C. Rijsbergen, Information Retrieval. 2nd ed., London: Butterworths, 1979.

Dr. Taoxin Peng He received his BSc degree in applied mathematics in 1982, his MSc degree in operational research in 1988, China, and his PhD in computer science from the University of Greenwich, London, UK in 2000.

He started his academic career as an Assistant Lecturer in China from 1982. He joined the Department of Artificial Intelligence at the University of Edinburgh, as a Research Associate in 1998. Since 1999, he became a Lecturer in the School of Computing at Edinburgh Napier University. His research findings have been published in both peer reviewed international conferences and journals. His current research interests include data quality, data cleaning, data mining and data warehousing.

Dr. Peng is a Fellow of Higher Education Academy (HEA), UK.

Mr. Lin Li received his BSc degree in computer science from Shenyang Aerospace University, China in 2004, and his MSc degree in software engineering from the University of Manchester, UK in 2006. Currently he is doing his PhD research study in School of Computing at Edinburgh Napier University, UK.

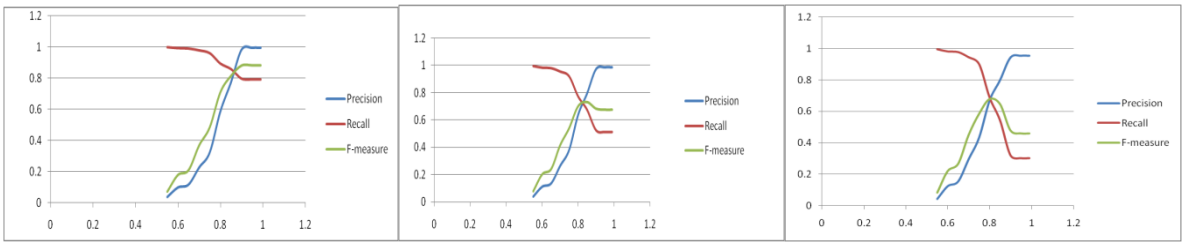
He has published several research papers. His research interests include data quality, data cleaning and software engineering.

Prof. Jessie Kennedy was born in UK. She received her BSc Honours degree in biology in 1980, and her MPhil degree in ecological database management from the University of Paisley, UK in 1983.

She has been with Edinburgh Napier University since 1986, where she has held the post of professor since 2000 and is currently Director of the Institute for Informatics and Digital Innovation. She has published widely with over 90 peer reviewed publications, has had over £1 million in research funding, has been programme chair, committee member and organiser of many international conferences and acts as reviewer for many national computer science funding bodies. She main research interests are in information visualization and database systems.

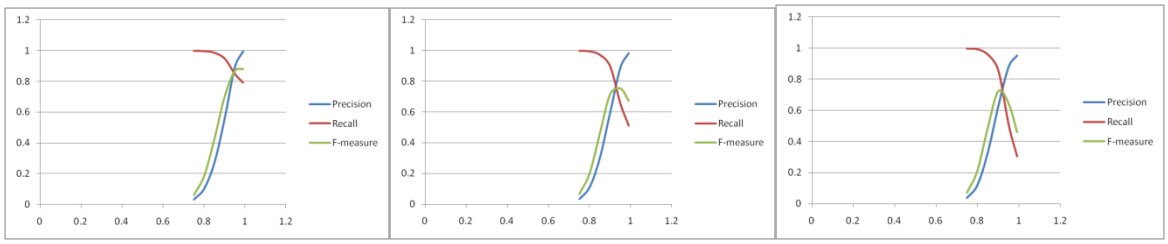
Professor Kennedy bas been a Member of EPSRC Peer Review College since 1996 and is a Fellow of the BCS, UK.

Levenshtein



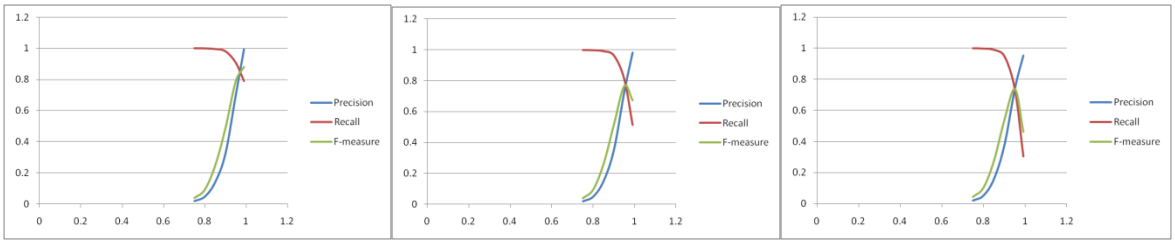
(a) Low Error Dataset (b) Medium Error Dataset (c) High Error Dataset

Jaro



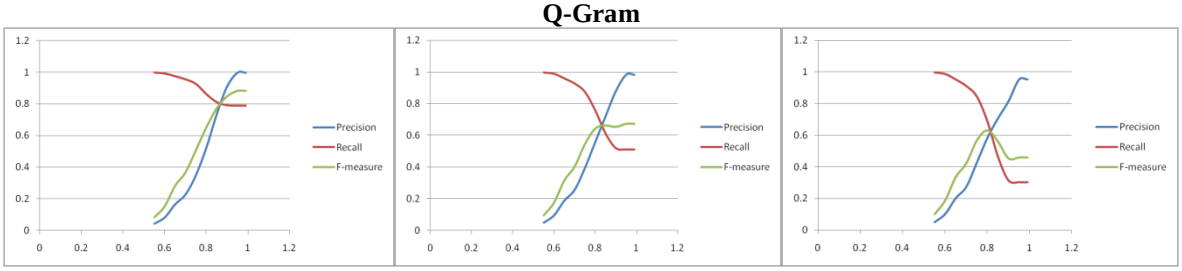
(a) Low Error Dataset (b) Medium Error Dataset (c) High Error Dataset

Jaro-Winkler



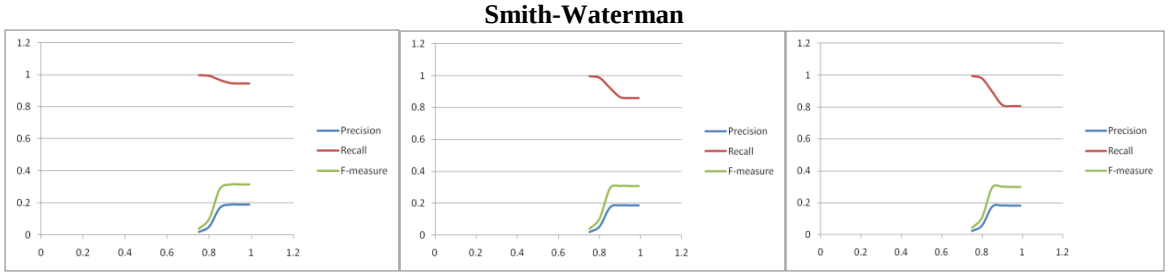
(a) Low Error Dataset (b) Medium Error Dataset (c) High Error Dataset

Q-Gram



(a) Low Error Dataset (b) Medium Error Dataset (c) High Error Dataset

Smith-Waterman



(a) Low Error Dataset (b) Medium Error Dataset (c) High Error Dataset

Fig. 1: Accuracy relative to the value of threshold on different datasets with different error rates